IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Snodgrass, Timothy E.
Title:	JTRS SCA CO-PROCESSOR ENGINE
Appl. No.:	10/789,609
Filing Date:	2/27/2004
Examiner:	Kimbleann C. Verdi
Art Unit:	2194
Confirmation Number:	9039

DECLARATION UNDER 37 CFR 1.131

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

I, Timothy E. Snodgrass, state and declare that:

1. I am the inventor of the claims currently pending in U.S. Patent Application No. 10/789,609 (hereinafter "the '609 application").

2. I understand that in an Office Action dated May 13, 2008, Claims were rejected as being unpatentable based in part on the use of Paulin et al., "Application of a Multi-Processor SoC Platform to High-Speed Packet Forwarding", Proceedings of the conference on Design, automation and test in Europe, p.30058, February 16-20, 2004 (hereinafter "Paulin").

3. I understand based on the information provided by the IEEE Computer Society, which is the publisher of the proceedings of the conference on Design, Automation and Test in Europe, that Paulin was disclosed at the conference on or after February 16, 2004.

4. At least before February 16, 2004, I conceived in the United States the ideas set forth in the pending claims of the '609 application. Such conception is evidenced by the attached letter and corresponding enclosed patent application document for filing pertaining to the subject matter of the present application.

5. After conceiving the ideas set forth in the pending claims of the '609 application, I engaged in diligence from prior to February 16, 2004 disclosure date of Paulin until the filing of the '609 application on February 27, 2004.

6. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are true, and further that these statements are made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the patent application or any patent issuing therefrom.

Date: AUGUST 12, 2008 By: Timothy E. Snodgrass
Timothy E. Snodgrass

FOLEY & LARDNER
ATTORNEYS AT LAW

FOLEY & LARDNER
VEREX PLAZA
150 EAST GILMAN STREET
MADISON, WISCONSIN 53703-1481
POST OFFICE BOX 1497
MADISON, WISCONSIN 53701-1497
608.257.5035 TEL
608.258.4258 FAX
www.foleylardner.com

February 12, 2004

CLIENT/MATTER NUMBER
047141-0300

Mr. Kyle Eppele
MS 124-323
Rockwell Collins
400 Collins Rd. NE
Cedar Rapids, IA 52498-0001

Ms. Sheila Mathews
MS 124-323
Rockwell Collins
400 Collins Rd. NE
Cedar Rapids, IA 52498-0001

Re: New U.S. Patent Application for: JOINT TACTICAL RADIO SYSTEM (JTRS) SOFTWARE COMPUTER ARCHITECTURE (SCA) CO-PROCESSOR
Our Ref. No.: 047141-0348
Your Ref. No.: 03CR254/KE

Dear Kyle and Sheila:

Per my recent e-mail sending you the electronic version of the application and formal drawings for the above-reference new patent application, enclosed please find paper copies of the same documents for you to prepare for filing.

Of course, if you have any questions, please do not hesitate to contact me.

Very truly yours,

Paul S. Hunter

PSH/drs
Enclosures

Atty. Dkt. No. 03CR254/KE

U.S. PATENT APPLICATION

for

JOINT TACTICAL RADIO SYSTEM (JTRS) SOFTWARE COMPUTER ARCHITECTURE (SCA) CO-PROCESSOR

by

Timothy E. Snodgrass

Express Mail Mailing Label _____
Date of Deposit _____
I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 C.F.R. §1.10 on the date indicated above and is addressed to the Mail Stop PATENT APPLICATION, Commissioner for Patents, PO Box 1450, Alexandria, Virginia 22313-1450.
Typed or printed name of person mailing paper or fee
(Signature of person mailing paper or fee)

JOINT TACTICAL RADIO SYSTEM (JTRS) SOFTWARE COMPUTER ARCHITECTURE (SCA) CO-PROCESSOR

FIELD OF THE INVENTION

[0001] The present invention relates to computer processing. More specifically, the present invention relates to a joint tactical radio system (JTRS) software computer architecture (SCA) co-processor.

BACKGROUND OF THE INVENTION

[0002] Communication middleware resides between client and server applications in distributed systems. Middleware simplifies application development by providing a uniform view of networks, protocols, and operating system layers. Object Request Brokers (ORBs), such as COBRA, DCOM, and Java RMI, eliminate many of the tedious, error-prone, and non-portable aspects of developing and maintaining applications programmed using mechanisms like sockets. In particular, ORBs automate common network programming tasks, such as object location, object activation, parameter (de)marshalling, socket and request demultiplexing, fault recovery, and security.

[0003] The United States Department of Defense has mandated compliance to the Joint Tactical Radio System (JTRS) Software Computer Architecture (SCA) for all new communication radios that operate between 2 MHz and 2000 MHz. The SCA further requires software applications to run under an operating system and that data be passed using COBRA middleware. CORBA (Common Object Request Broker Architecture) provides platform-independent programming interfaces and models for portable distributed-oriented computing applications.

[0004] Conventional CORBA implementations have limitations that make it hard for them to support performance-sensitive applications effectively. For example, conventional ORBs like COBRA support a limited number of statically configured protocols, often just GIOP (General Inter-ORB Protocol) / IIOP (Internet Inter-ORB Protocol) over TCP/IP. Further, conventional ORBs do not allow applications to configure key protocol policies and properties, such as peak virtual circuit bandwidth or cell pacing rate. Moreover, conventional ORBs do not support simultaneous use of multiple inter-ORB messaging or transport protocols. Yet further, conventional ORBs

have limited or no support for specifying and enforcing real-time protocol requirements across a backplane, network, or Internet end-to-end.

[0005] Another significant disadvantage of commercial middleware, such as COBRA, and operating systems available today is they are much too slow to allow use in power-constrained applications, such as the Objective Force Warrior, munitions and Unattended Ground Sensors (JTRS Cluster "5"). By adding sufficient processing power to overcome the latency caused by the operating system and middleware, the resulting heat and battery life reductions make compliance with JTRS SCA requirements virtually impossible. For non-battery powered applications, such as JTRS Cluster 1, there is a tremendous recurring cost and heat burden that the program cannot reach. A large part of the performance demand is required performance because of the excessive heat dissipated by numerous 1000 MIP processors.

[0006] One attempt to resolve some of the limitations of middleware, such as COBRA, is implementing a pluggable protocol framework. A pluggable protocol framework refers to using customized ORB interfaces that allow data to be passed without going through the operating system. The pluggable protocols are software applications designed using a standard protocol configuration to allow custom messaging and transport protocols to be configured flexibly. Nevertheless, software pluggable protocols are similar in speed to the middleware application and they may not be portable across platforms.

[0007] Thus, there is a need to improve the speed by which data is communicated in a communication system by avoiding software applications, such as the operating system and middleware. Further, there is a need to improve latency in SCA compliant software solutions such that more efficient processors can be used. Even further, there is a need for a joint tactical radio system (JTRS) software computer architecture (SCA) co-processor that implements essential services to a waveform application.

SUMMARY OF THE INVENTION

[0008] The present invention includes embodiments having an application specific integrated circuit (ASIC) or field programmable gate arrays (FPGA) that implements essential services that an operating system and CORBA provide to a waveform application. The embodiments reduce latency in middleware and the operating system such that more efficient processors with less total capability can be used. For example, an exemplary embodiment relates to an apparatus that implements services for a waveform application. The apparatus includes an object request broker that marshals data from the waveform application for communication and an object request broker interface that communicates the marshaled data using a memory pool. At least a portion of the object request broker is implemented in hardware and at least a portion of the object request broker interface is implemented in hardware.

[0009] Another exemplary embodiment relates to a method of marshalling transactions for waveform application communications using a CORBA (Common Object Request Broker Architecture) broker. The method includes marshalling data from a waveform application in a first communication device and interfacing the marshaled data with a second communication device using a memory pool. At least a portion of the marshalling operation is implemented in hardware and at least a portion of the interfacing operation is implemented in hardware.

[0010] Still another exemplary embodiment relates to a system for a joint tactical radio system (JTRS) compliant device that provides communication at low power requirements. The system includes a hardware-implemented object request broker (ORB) that marshals data from a waveform application, a pluggable protocol interface that communicates the marshaled data from the hardware-implemented ORB, and a memory pool that communicates data from the pluggable protocol interface directly and without transport overhead. At least a portion of the pluggable protocol interface is implemented in hardware.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The exemplary embodiments will be described with reference to the accompanying drawings, wherein like numerals denote like elements; and

[0012] Fig. 1 is a diagram of a communication apparatus having an application specific integrated circuit (ASIC) in accordance with an exemplary embodiment;

[0013] Fig. 2 is a functional view of an architecture for power restricted applications in accordance with a conventional system;

[0014] Fig. 3 is a functional view of an architecture for power restricted applications included in the communication apparatus of Fig. 1 in accordance with an exemplary embodiment; and

[0015] Fig. 4 is an implementation view of an architecture for power restricted applications included in the communication apparatus of Fig. 1 in accordance with an exemplary embodiment.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS

[0016] Fig. 1 illustrates a communication apparatus 10 having a battery 12. In an exemplary embodiment, the battery 12 has approximately 30-100 watt-hours in power. As such, if the communication apparatus 10 had to operate in an application for a week without a re-charge, it would be limited to a draw of 0.2 to 0.6 watts (time averaged power).

[0017] An application specific integrated circuit (ASIC) 14 is included in the communication apparatus 10. The ASIC 14 is programmable by software stored in the communication apparatus 10. The ASIC 14 implements various services that an operating system and an Object Request Broker (ORB) otherwise provide to a waveform application. Implementing services as such reduces operating system and middleware latency. Lower latency allows the use of more efficient processors based on MIPS per milliwatt. Such more efficient processors enable SCA compliant software solutions for battery powered radios, missles, small unmanned vehicles, and unmanned ground sensors.

[0018] Fig. 2 illustrates a functional view of an architecture for power restricted applications according to known systems. A waveform application 22 communicates with a waveform application 40 using a variety of functions.

[0019] Conventional systems utilize a user data protocol (UDP) / transmission control protocol (TCP) interface 28 or a proprietary connections interface 30. The UDP / TCP interface 28 communicates data to an Ethernet 34 and a SCMP (Snuggly Coupled Multi Processor) interface 36. In like fashion, the ORB 42 provides data to a user data protocol (UDP) / transmission control protocol (TCP) interface 48 and a proprietary connections interface 50. The UDP / TCP interface 28 communicates data to an Ethernet 54 and a SCMP 56.

[0020] Fig. 3 illustrates a functional view of an architecture for power restricted applications included in the communication apparatus 10 of Fig. 1. A waveform application 22 communicates with a waveform application 40 using a variety of functions. Additional, fewer, or different functions may be performed in the communication process, depending on the particular embodiment.

[0021] The waveform application 22 uses an Object Request Broker (ORB) 24 to marshal data for communication to the waveform application 42. Similarly, waveform application 42 uses an ORB 44 to marshal data for communication to the waveform application 22.

[0022] According to an exemplary embodiment, in contrast to the conventional communication examples described with reference to Fig. 2, which have extensive delays in transporting data, custom ORB interfaces 26 and 46 can directly transport data into the process space of another process. The custom ORB interfaces 26 and 46 communicate via a multi-port memory pool 38. The memory pool 38 has the ability to appear in each process space and thus can be used to pass data directly and without transport overhead such as is incurred using TCP/IP.

[0023] Conventional systems have tried to achieve such efficiency and performance in several ways. The proprietary connections interfaces 30 and 50 (Fig. 1) are examples of an operating system (OS) communicating via a unique, non-portable operating system interface. The proprietary connections interfaces 30 and 50 cannot communicate with the operating system of another vendor. Such a restriction is unsuitable for JTRS requirements. The UDP/TCP interfaces 28 and 48 is another attempt

for industry to communicate from one process space to another via operating system transfers. It, however, incurs very large overhead for simple transfers because it traverses the protocol stack and even supplies guaranteed return receipts for each transaction (TCP). While TCP may be reasonable when communicating New York to Paris, it is an absurd overhead to carry when one application is communicating to another application on the same processor or to the processor soldered on the same circuit card. The Ethernets 34 and 54 illustrate the most common form of the conventional art, that of communicating between different applications via Ethernet which is even more inefficient because it layers Ethernet protocol on top of TCP/IP.

[0024] Fig. 4 illustrates an implementation view of an architecture for power restricted applications included in the communication apparatus 10 of Fig. 1. In Fig. 3, the power wasting middleware has been reduced to hardware and the functions that the operating system (OS) supplied which caused such delays (and demanded increased throughput) have been replaced by enabling the waveform application 22 to communicate directly with the waveform application 40 using ORBs 58 and 59 implemented in hardware and pluggable protocol interfaces 60 and 62. The pluggable protocol interfaces 60 and 62 provide a standardized interface as defined by the OMG (Object Management Group, an international standards making consortium).

[0025] The middleware and operating system functionality that is inefficient for software to implement is carried out in hardware. Alternatively, microprocessors are used that talk CORBA marshaled data natively. As such, the processor is relieved from executing a software application to arrange its bits in the right order. Because the ORB is intimately connected with the OS and the way the OS communicates into another "process space," just implementing the ORB in hardware is not sufficient because in many applications it is the slowness of the OS implementing the protocol stack that requires the excessive throughput. Thus, not all the OS or ORB is implemented in hardware, but rather that the portions that are impacting performance are implemented in hardware. For example, the OS protocol stack can be implemented in hardware. The OS protocol stack is portable and a standard way of interfacing. Data marshaling functions and logic and data functions that consume excessive processor throughput can also be implemented in hardware.

[0026] It is understood that although the detailed drawings and specific examples describe exemplary embodiments of a joint tactical radio system (JTRS)

software computer architecture (SCA) co-processor, they are for purposes of illustration only. The exemplary embodiments are not limited to the precise details and descriptions described herein. For example, although particular devices and structures are described, other devices and structures could be utilized according to the principles of the present invention. Various modifications may be made and the details disclosed without departing from the spirit of the invention as defined in the following claims

WHAT IS CLAIMED IS:

1. An apparatus that implements services for a waveform application, the apparatus comprising:

an object request broker that marshals data from the waveform application for communication, wherein at least a portion of the object request broker is implemented in hardware; and

an object request broker interface that communicates the marshaled data using a memory pool, wherein at least a portion of the object request broker interface is implemented in hardware.

2. The apparatus of claim 1, wherein the apparatus is an application specific integrated circuit (ASIC).

3. The apparatus of claim 1, wherein the apparatus is a field programmable gate array (FPGA).

4. The apparatus of claim 1, wherein the object request broker interface comprises a pluggable protocol interface.

5. The apparatus of claim 1, wherein the object request broker interface comprises a custom interface.

6. The apparatus of claim 1, wherein the object request broker is a CORBA (Common Object Request Broker Architecture) broker.

7. The apparatus of claim 1, wherein the memory pool comprises a multi-port memory pool.

8. The apparatus of claim 1, wherein the at least a portion of the object request broker that is implemented in hardware comprises logic and data formatting functions that are determined to consume excessive processor throughput for a software application.

9. The apparatus of claim 1, wherein the at least a portion of the object request broker interface that is implemented in hardware comprises an operating

system protocol stack.

10. A method of marshalling transactions for waveform application communications using a CORBA (Common Object Request Broker Architecture) broker, the method comprising:

marshalling data from a waveform application in a first communication device, wherein at least a portion of the marshalling operation is implemented in hardware; and

interfacing the marshaled data with a second communication device using a memory pool, wherein at least a portion of the interfacing operation is implemented in hardware.

11. The method of claim 10, wherein the at least a portion of the marshalling operation that is implemented in hardware comprises logic and data formatting functions that are determined to consume excessive processor throughput for a specific software application.

12. The method of claim 10, wherein the at least a portion of the interfacing operation that is implemented in hardware comprises an operating system protocol stack.

13. The method of claim 11, wherein the hardware comprises an application specific integrated circuit (ASIC).

14. The method of claim 11, wherein the hardware comprises a field programmable gate array (FPGA).

15. A system for a joint tactical radio system (JTRS) compliant device that provides communication at low power requirements, the system comprising:

a hardware-implemented object request broker (ORB) that marshals data from a waveform application;

a pluggable protocol interface that communicates the marshaled data from the hardware-implemented ORB, wherein at least a portion of the pluggable protocol interface is implemented in hardware; and

a memory pool that communicates data from the pluggable protocol interface directly and without transport overhead.

16. The system of claim 15, wherein the at least a portion of the pluggable protocol interface that is implemented in hardware comprising logic and data formatting functions of the ORB that are determined to consume excessive processor throughput for a specific software application and an interface to a shared memory pool.

17. The system of claim 16, wherein the hardware comprises an application specific integrated circuit (ASIC).

18. The system of claim 16, wherein the hardware comprises a field programmable gate array (FPGA).

19. The system of claim 15, wherein the JTRS compliant device is in an unmanned craft.

20. The system of claim 15, wherein the JTRS compliant device is a battery powered radio.

JOINT TACTICAL RADIO SYSTEM (JTRS) SOFTWARE COMPUTER ARCHITECTURE (SCA) CO-PROCESSOR

ABSTRACT OF THE DISCLOSURE

A joint tactical radio system (JTRS) software computer architecture (SCA) apparatus that implements services for a waveform application by reducing latency in middleware and the operating system such that more power efficient processors can be used. The apparatus includes an object request broker that marshals data from the waveform application for communication and an object request broker interface that communicates the marshaled data using a memory pool and select functions of an Operating System. At least a portion of the object request broker is implemented in hardware and at least a portion of the object request broker interface is implemented in hardware and at least a portion of the Operating System is implemented in hardware.

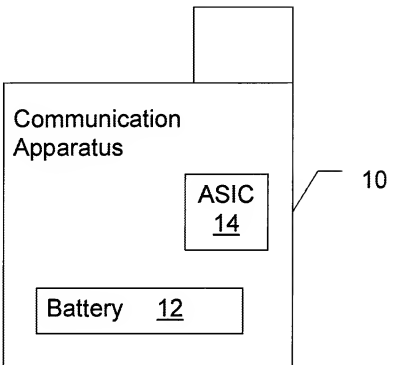

FIG. 1

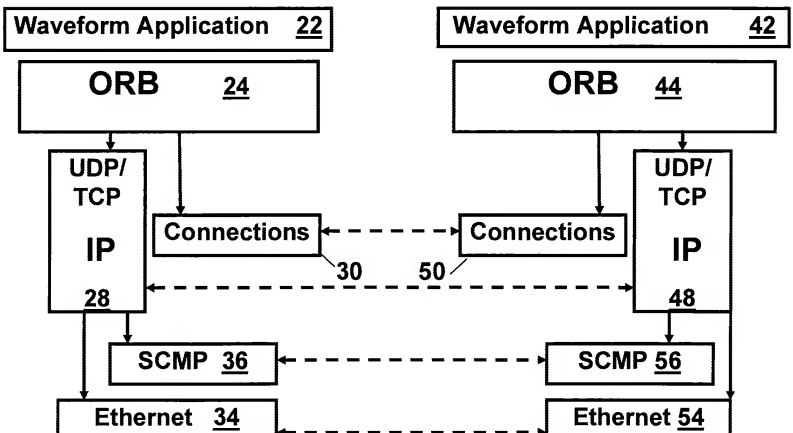

FIG. 2

PRIOR ART

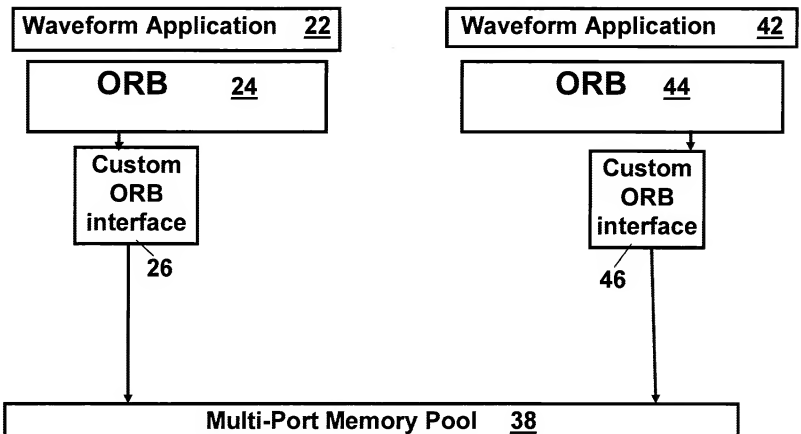
Waveform Application 22
ORB 24
Custom ORB interface
26
Waveform Application 42
ORB 44
Custom ORB interface
46
Multi-Port Memory Pool 38

FIG. 3

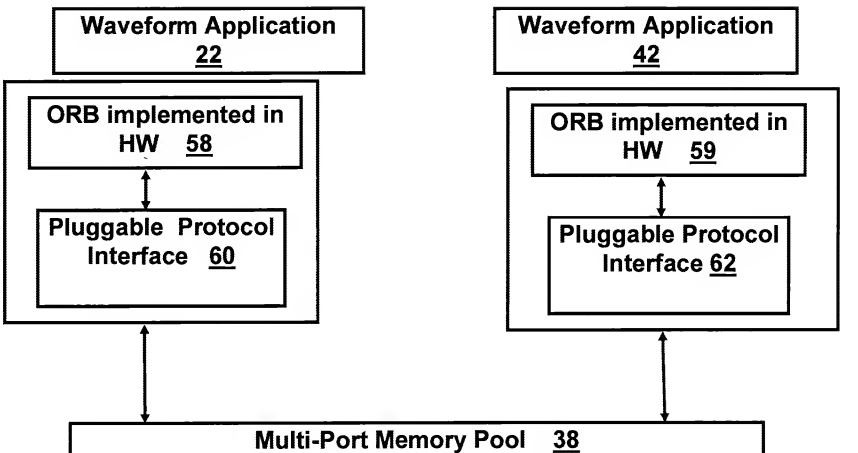
Waveform Application
22
Waveform Application
42
ORB implemented in
HW 58
ORB implemented in
HW 59
Pluggable Protocol
Interface 60
Pluggable Protocol
Interface 62
Multi-Port Memory Pool 38

FIG. 4